Pergamon

Computers Math. Applic. Vol. 31, No. 4/5, pp. 105–112, 1996

0898-1221/96 $15.00 + 0.00

0898-1221(95)00222-7

General Linear Methods

J. C. BUTCHER
Department of Mathematics, The University of Auckland
Private Bag 92019, Auckland, New Zealand
butcher@math.auckland.ac.nz

Abstract—General linear methods were originally introduced to provide a unified theory of consistency, stability and convergence for a large class of numerical methods for ordinary differential equations. We survey the use of this formulation and discuss the meaning of order of accuracy for these methods. In the search for new practical algorithms, we consider the special case of "DIMSIM" methods.

Keywords—General linear methods, Differential equations.

1. INTRODUCTION

General linear methods were originally proposed [1] as a unifying framework for studying stability, consistency and convergence for a wide variety of methods. Not only does their formulation include both Runge-Kutta and linear multistep methods in a natural way, but also allows for various generalizations of these methods proposed by a number of authors [2–4]. In Section 2, we will introduce the formulation for these methods now in common use, and show how a number of special method types can be represented within this formulation. In Section 3, we will review the basic definitions of stability, consistency and convergence and indicate the relationship between them. In Section 4, we discuss the concept of order for general linear methods and show through some examples how it can be analysed in practice. Section 5 is devoted to a special class of general linear method known as diagonally-implicit multistage integration methods (DIMSIMs) [5] which seem to have the best prospects for practical implementation.

In Section 6, we will consider some type 1 DIMSIMs, constructed to have the same stability regions as explicit Runge-Kutta methods of the same order. In addition to the basic methods, we will present details of interpolation, error estimation and stepsize changing schemes designed to be used with them.

2. GENERAL LINEAR METHODS

The two traditional classes of numerical methods for ordinary differential equations have always been studied separately. For linear multistep methods, the difficult questions have always been associated with the stability of the sequence of approximations produced whereas accuracy questions are relatively simple. For Runge-Kutta methods, stability is a very straightforward matter but accuracy questions are exceedingly complicated. In the years between 1960 and 1970, a number of methods were introduced which fall between these traditional classes. For these new methods each of the existing theories became, to some extent, unnatural.

General linear methods were introduced to provide a unifying theory of the basic questions of consistency, stability and convergence. Later they were used as a framework for studying accuracy questions and later the phenomena associated with nonlinear convergence. In the notation introduced in [6], a general linear method is represented by a partitioned $(s+r) \times (s+r)$ matrix

$$\begin{bmatrix} A & U \\ B & V \end{bmatrix}.$$

For an N-dimensional differential equation system $y'(x) = f(y(x))$, the method characterized in this way makes use of r quantities as output from each step and input to the next step. Denote these quantities, as computed at the end of step number n by $y_1^{(n)}, y_2^{(n)}, \ldots, y_r^{(n)}$ and write these together as a vector in rN dimensions given by

$$y^{(n)} = \begin{bmatrix} y_1^{(n)} \\ y_2^{(n)} \\ \vdots \\ y_r^{(n)} \end{bmatrix}.$$

Similarly, denote s stage values used in the computation of the step by Y_1, Y_2, ..., Y_s and write these together as a vector in sN dimensions given by

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_s \end{bmatrix}.$$

If I is the $N \times N$ unit matrix and $f(Y)$ denotes an sN-dimensional vector given by

$$\begin{bmatrix} f(Y_1) \\ f(Y_2) \\ \vdots \\ f(Y_s) \end{bmatrix},$$

then the quantities $y^{(n)}, Y, f(Y)$ and $y^{(n-1)}$ are related by

$$\begin{bmatrix} Y \\ y^{(n)} \end{bmatrix} = \begin{bmatrix} A \otimes I & U \otimes I \\ B \otimes I & V \otimes I \end{bmatrix} \begin{bmatrix} hf(Y) \\ y^{(n-1)} \end{bmatrix}.$$

To see how existing methods can be represented in this formulation, consider the example of the classical fourth order Runge-Kutta method with tableau

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

for which the general linear formulation is in terms of the matrix

$$\left[\begin{array}{cccc|c} 0 & 0 & 0 & 0 & 1 \\ \frac{1}{2} & 0 & 0 & 0 & 1 \\ 0 & \frac{1}{2} & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ \hline \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} & 1 \end{array}\right].$$

Similarly, for the Adams-Bashforth method,

$$y_n = y_{n-1} + h\left(\frac{3}{2}f(y_{n-1}) - \frac{1}{2}f(y_{n-2})\right),$$

the general linear formulation is

$$\left[\begin{array}{c|cc} 0 & 1 & \frac{3}{2} & -\frac{1}{2} \\ \hline 0 & 1 & \frac{3}{2} & -\frac{1}{2} \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right].$$

Note that Runge-Kutta methods always have $r = 1$, since only one quantity is passed from one step to the next. On the other hand linear multistep method have $s = 1$ because the function f is evaluated only once in each step.

A final example is for the hybrid method which evaluates an approximate solution $y^*_{n-(1/2)}$ midway through step number n, a "predicted" approximate solution y^*_n at the end of the step and finally a "corrected" approximation y_n; using the formulas

$$\begin{aligned} y^*_{n-1/2} &= y_{n-2} + h\left(\frac{9}{8}f(y_{n-1}) + \frac{3}{8}f(y_{n-2})\right), \\ y^*_n &= \frac{28}{5}y_{n-1} - \frac{23}{5}y_{n-2} + h\left(\frac{32}{15}f(y^*_{n-1/2}) - 4f(y_{n-1}) - \frac{26}{15}f(y_{n-2})\right), \\ y_n &= \frac{32}{31}y_{n-1} - \frac{1}{31}y_{n-2} + h\left(\frac{5}{31}f(y^*) + \frac{64}{93}f(y^*_{n-1/2}) + \frac{4}{31}f(y_{n-1}) - \frac{1}{93}f(y_{n-2})\right). \end{aligned}$$

As a general linear method this can be written as

$$\left[\begin{array}{ccc|cccc} 0 & 0 & 0 & 0 & 1 & \frac{9}{8} & \frac{3}{8} \\ \frac{32}{15} & 0 & 0 & \frac{28}{5} & -\frac{23}{5} & -4 & -\frac{26}{15} \\ \frac{64}{93} & \frac{5}{31} & 0 & \frac{32}{31} & -\frac{1}{31} & \frac{4}{31} & -\frac{1}{93} \\ \hline \frac{64}{93} & \frac{5}{31} & 0 & \frac{32}{31} & -\frac{1}{31} & \frac{4}{31} & -\frac{1}{93} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}\right].$$

3. CONSISTENCY, STABILITY AND CONVERGENCE

Since the quantities passed from step to step may represent approximations to y values, hy' values or arbitrary linear combinations of such quantities, consistency must be defined in a rather general manner. Two r dimensional vectors are introduced, written as u and v and referred to as the "preconsistency" and "consistency" vectors, respectively. The significance of these vectors is that $y_i^{(n)}$ is intended to approximate $u_i y(x_n) + hv_i y'(x_n)$. If e denotes the s dimensional vector

with each component equal to 1, then the full preconsistency and consistency conditions are given by

$$Vu = u, \qquad Uu = e, \qquad Be + Vv = u + v.$$

If these conditions are satisfied, then the numerical method is at least capable of propagating exact solutions to the trivial initial value problem $y' = 1$.

Stability is defined, as a generalisation of the definition for linear multistep methods, as the requirement that V is a power-bounded matrix.

We will not formally define convergence but it can be described informally as the ability of a numerical method to produce approximations which converge in the limit as the number of steps tends to infinity (as the stepsize converges to zero in an appropriate manner) to the exact solution with the r values scaled according to the components of the preconsistency vector.

Just as for classical linear multistep methods, the equivalence of convergence to consistency plus stability can be established. Using a slightly different but equivalent formulation than that given in this paper, this was proved in [1].

4. ORDER OF GENERAL LINEAR METHODS

As for all multivalue methods, some sort of starting procedure is required before the first step can be carried out. The difference between these methods and linear multistep methods is the generality of the quantities that general linear methods pass from one step to the next. Hence, it is appropriate to allow for a starting method which computes a variety of possible quantities $y_1^{(0)}, y_2^{(0)}, \ldots, y_r^{(0)}$ for input to the first step of integration.

A convenient interpretation of what the method is required to do can be obtained by regarding a similar starting operation to be applied to every point on the exact trajectory. Thus, alongside the exact solution, a collection of r further "trajectories" can be found. The purpose of the numerical method can be thought of as maintaining accurate approximations to these associated trajectories. For a given starting method, let $\hat{y}_1^{(n)}, \hat{y}_2^{(n)}, \ldots, \hat{y}_r^{(n)}$ denote the result of applying this starting method at $y(x_n)$. Relative to the starting procedure, a method is said to have order p if $y_1^{(0)} = \hat{y}_1^{(0)}, y_2^{(0)} = \hat{y}_2^{(0)}, \ldots, y_r^{(0)} = \hat{y}_r^{(0)}$ implies $y_1^{(1)} = \hat{y}_1^{(1)} + O(h^{p+1}), y_2^{(1)} = \hat{y}_2^{(1)} + O(h^{p+1}), \ldots, y_r^{(1)} = \hat{y}_r^{(1)} + O(h^{p+1})$.

In each of the examples given in the previous section, it is easy to verify that the order obtained in this way is the same as the order defined in a natural and classical manner. However, it is interesting that for Runge-Kutta methods, it is possible to obtain a value of the order higher than the classical value, because the starting method need not be the identity operation, as it is usually taken to be for a Runge-Kutta method. In particular, it is possible to obtain an order 5 in this more general sense with a 5 stage explicit Runge-Kutta method [7], whereas this is not possible under the classical definition of order.

5. DIAGONALLY IMPLICIT MULTISTAGE INTEGRATION METHODS

These methods, also known as DIMSIMs were introduced [5] in the hope of identifying special subfamilies, within the large class of general linear methods, which seem to have greatest prospects for practical use. To lower the cost of implementation, it seems to be desirable to restrict A to lower triangular form, as for an explicit or a diagonally implicit Runge-Kutta method. Because of computational difficulties related to low stage-order, it seems desirable to require that the stage order should be equal to, or at least be not much lower than, the full order of a method. To obtain good stability at zero, it seems to be desirable that the matrix V have a simple structure; it should have a spectrum $\{0, 1\}$ and should preferably be of rank 1. Finally, the very free and general order conditions discussed in Section 4 should be restricted to a requirement that the

starting values should be given by a weighted Taylor expansion. That is the approximations at the start of step number n should be of the form

$$y_i^{(n-1)} = \alpha_{i0} y(x_{n-1}) + h\alpha_{i1} y'(x_{n-1}) + \cdots + h^p \alpha_{ip} y^{(p)}(x_{n-1}) + O(h^{p+1}), \qquad i = 1, 2, \ldots, r,$$

with similar approximations found at the end of the step but with x_{n-1} replaced by x_n.

Methods that satisfy these requirements will be referred to as DIMSIMs.

Amongst this still large family, we distinguish four subfamilies known as types.

Type 1. These are intended for nonstiff problems on a sequential computer and A is required to be of lower triangular form and to have zeros on its diagonal.

Type 2. These are for stiff problems on a sequential computer and A is assumed to be of lower triangular form with constants on the diagonal.

Type 3. These are for nonstiff problems in a parallel environment and A is required to be the zero matrix.

Type 4. For stiff problems on a parallel computer with A required to be a diagonal matrix. For simplicity, the diagonal elements will be assumed to be equal.

Examples are known of methods in each of these types at least for low orders.

6. THREE TYPE 1 METHODS

We discuss here only three methods. They are all of type 1 and have orders 1, 2, and 3, respectively. The order 1 method is the classical method of Euler, written in DIMSIM form as

$$\left[\begin{array}{c|c} 0 & 1 \\ \hline 1 & 1 \end{array}\right].$$

The method of order 2 was given in [5] and is interesting in that it has exactly the same stability region as for standard second order 2 stage Runge-Kutta methods but has stage order also equal to 2. The method is

$$\left[\begin{array}{cc|cc} 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ \hline \frac{5}{4} & \frac{1}{4} & \frac{1}{2} & \frac{1}{2} \\ \frac{3}{4} & -\frac{1}{4} & \frac{1}{2} & \frac{1}{2} \end{array}\right].$$

The order 3 method [8] has a similar property: its stage order is 3 and yet it has the same stability region as for a 3 stage third order Runge-Kutta method. Its defining matrix is

$$\left[\begin{array}{ccc|ccc} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ \frac{1}{4} & 1 & 0 & 0 & 0 & 1 \\ \hline \frac{5}{4} & \frac{1}{3} & \frac{1}{6} & -\frac{2}{3} & \frac{4}{3} & \frac{1}{3} \\ \frac{35}{24} & -\frac{1}{3} & \frac{1}{8} & -\frac{2}{3} & \frac{4}{3} & \frac{1}{3} \\ -\frac{17}{12} & 0 & \frac{1}{12} & -\frac{2}{3} & \frac{4}{3} & \frac{1}{3} \end{array}\right].$$

As they stand, these three methods are of little value because they do not have built in error estimates, interpolated output, or stepsize and order changing procedures. These features will now be added to these methods so that, within the limitations implied by their low orders, they could form the basis of a flexible ordinary differential equation solver. In principle, all the attributes present in these methods could be extended to higher orders.

To obtain local error estimates, we first discuss the nature of the local truncation error. If it is assumed that the starting vector for a step of the order 1 method is

$$y^{(n-1)} = [y(x_{n-1}) + O(h^3)],$$

then the outgoing vector is found to be

$$y^{(n)} = \left[y(x_n) - \frac{h^2}{2}y''(x_n) + O(h^3)\right].$$

Similarly, for the order 2 method with incoming vector

$$y^{(n-1)} = \begin{bmatrix} y(x_{n-1}) + O(h^4) \\ y(x_{n-1}) - hy'(x_{n-1}) + \frac{h^2}{2}y''(x_{n-1}) - \frac{h^3}{4}y'''(x_{n-1}) + O(h^4) \end{bmatrix},$$

the outgoing vector is

$$y^{(n)} = \begin{bmatrix} y(x_n) - \frac{h^3}{6}y'''(x_n) + O(h^4) \\ y(x_n) - hy'(x_n) + \frac{h^2}{2}y''(x_n) - \left(\frac{h^3}{4} + \frac{h^3}{6}\right)y'''(x_n) + O(h^4) \end{bmatrix}.$$

Finally, for the order 3 method with incoming vector

$$y^{(n-1)} = \begin{bmatrix} y(x_{n-1}) + O(h^5) \\ y(x_{n-1}) - \frac{h}{2}y'(x_{x-1}) + \frac{h^2}{8}y''(x_{n-1}) + \frac{h^3}{48}y'''(x_{n-1}) - \frac{h^4}{48}y^{(iv)}(x_{n-1}) + O(h^5) \\ y(x_{n-1}) - \frac{h}{4}y'(x_{n-1}) + \frac{h^3}{24}y'''(x_{n-1}) - \frac{h^4}{48}y^{(iv)}(x_{n-1}) + O(h^5) \end{bmatrix},$$

we find the outgoing vector to be

$$y^{(n)} = \begin{bmatrix} y(x_n) - \frac{h^4}{24}y^{(iv)}(x_n) + O(h^5) \\ y(x_n) - \frac{h}{2}y'(x_n) + \frac{h^2}{8}y''(x_n) + \frac{h^3}{48}y'''(x_n) - \left(\frac{h^4}{48} + \frac{h^4}{24}\right)y^{(iv)}(x_n) + O(h^5) \\ y(x_n) - \frac{h}{4}y'(x_n) + \frac{h^3}{24}y'''(x_n) - \left(\frac{h^4}{48} + \frac{h^4}{24}\right)y^{(iv)}(x_n) + O(h^5) \end{bmatrix}.$$

Thus the local truncation errors for each component estimated can be found if it is possible to obtain good approximations to $(h^2/2)y''(x_n)$, $(h^3/6)y'''(x_n)$, $(h^4/24)y^{(iv)}(x_n)$, respectively.

This can be accomplished by evaluating further linear combinations of $hf(Y_1), hf(Y_2), \ldots, hf(Y_s)$ and $y_1^{(n-1)}, y_2^{(n-1)}, \ldots, y_r^{(n-r)}$ so that the error estimates take the form

$$h\left(\widehat{b}_1 f(Y_1) + \widehat{b}_2 f(Y_2) + \cdots + \widehat{b}_s f(Y_s)\right) + \left(\widehat{v}_1 y_1^{(n-1)} + \widehat{v}_2 y_2^{(n-1)} + \cdots + \widehat{v}_r y_r^{(n-1)}\right).$$

Suitable vectors $\widehat{b} = [\widehat{b}_1, \widehat{b}_2, \ldots, \widehat{b}_s]^{\mathsf{T}}$ and $\widehat{v} = [\widehat{v}_1, \widehat{v}_2, \ldots, \widehat{v}_r]^{\mathsf{T}}$ can now be given. Note that in the case of order 1, it is necessary to anticipate the subsequent computation of the derivative

in the single stage of the following step so that, for this method we will now assume $s = 2$ and $r = 1$, so that the method is now defined by the matrix

$$\left[\begin{array}{cc|c} 0 & 0 & 1 \\ 1 & 0 & 1 \\ \hline 1 & 0 & 1 \end{array}\right].$$

For order 1, we have

$$\widehat{b} = \begin{bmatrix} -\frac{1}{2} \\ \frac{1}{2} \end{bmatrix}, \quad \widehat{v} = [0],$$

for order 2,

$$\widehat{b} = \begin{bmatrix} -\frac{1}{2} \\ \frac{1}{6} \end{bmatrix}, \quad \widehat{v} = \begin{bmatrix} \frac{1}{3} \\ -\frac{1}{3} \end{bmatrix},$$

and for order 3,

$$\widehat{b} = \begin{bmatrix} -\frac{1}{12} \\ -\frac{1}{3} \\ \frac{1}{6} \end{bmatrix}, \quad \widehat{v} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}.$$

To obtain dense output as well as variable stepsize, it is possible to use the incoming approximations $y^{(n-1)}$ and the stage derivatives to obtain approximations to a sequence of scaled derivatives $\widetilde{y}_0 \approx y(x_n), \widetilde{y}_1^{(n)} \approx hy'(x_n), \ldots, \widetilde{y}_p^{(n)} \approx h^p y^{[p]}(x_n)$ instead of the vector $y^{(n)}$. A change in h value can be achieved by rescaling the derivatives. The matrices, which we will denote $\widetilde{B}$ and $\widetilde{V}$, respectively, to carry out these requirements will now be given.

For order 1,

$$\widetilde{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \widetilde{V} \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

for order 2,

$$\widetilde{B} = \begin{bmatrix} \frac{5}{4} & \frac{1}{4} \\ 0 & 1 \\ -1 & 1 \end{bmatrix}, \quad \widetilde{V} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

and for order 3,

$$\widetilde{B} = \begin{bmatrix} \frac{5}{4} & \frac{1}{3} & \frac{1}{6} \\ 0 & 0 & 1 \\ 1 & -4 & 3 \\ 4 & -8 & 4 \end{bmatrix}, \quad \widetilde{V} = \begin{bmatrix} -\frac{1}{3} & \frac{4}{3} & \frac{1}{3} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

7. CONCLUSIONS AND FURTHER DEVELOPMENTS

General linear methods, especially those of the DIMSIM variety, seem to have considerable promise as alternatives to existing methods. Although numerical results are not included in this paper, preliminary testing shows these methods to be effective for a wide range of problems including stiff ordinary differential equations and differential-algebraic equations.

Many of the ideas and results found on DIMSIM methods have come out of collaborations with Jackiewicz and with Chartier. In the work with Jackiewicz, the order achievable for Types 1 and 2, together with implementation details, are being extended [9–10]. In the work with Chartier [11], Type 4 methods, suitable for both stiff problems and problems with algebraic constraints in a parallel computing environment are being investigated.

REFERENCES

1. J.C. Butcher, On the convergence of numerical solutions to ordinary differential equations, *Math. Comp.* **20**, 1–10 (1966).
2. C.W. Gear, Hybrid methods for inital value problems in ordinary differential equations, *SIAM J. Numer. Anal.* **2**, 69–86 (1965).
3. W.B. Gragg and H.J. Stetter, Generalized multistep predictor-corrector methods, *J. Assoc. Comput. Mach.* **11**, 188–209 (1964).
4. J.C. Butcher, A modified multistep method for the numerical integration of ordinary differential equations, *J. Assoc. Comput. Mach.* **12**, 124–135 (1965).
5. J.C. Butcher, Diagonally implicit multi-stage integration methods, *Appl. Numer. Math.* **11**, 347–363 (1993).
6. K. Burrage and J.C. Butcher, Non-linear stability of a general class of differential equation methods, *BIT* **20**, 185–203 (1980).
7. J.C. Butcher, The effective order of Runge-Kutta methods, *Lecture Notes in Mathematics* **109**, 133–139 (1969).
8. J.C. Butcher and Z. Jackiewicz, The construction of diagonally implicit general linear methods of type 1 and 2 for ordinary differential equations (in preparation).
9. J.C. Butcher and Z. Jackiewicz, Diagonally implicit general linear methods for ordinary differential equations, *BIT* **33**, 452–472 (1993).
10. J.C. Butcher and Z. Jackiewicz, The implementation of diagonally implicit multistage integration methods for ordinary differential equations (in preparation).
11. J.C. Butcher and P. Chartier, Parallel general linear methods for stiff ordinary differential and differential algebraic equations, *Appl. Numer. Math.* **17**, 213–222 (1995).